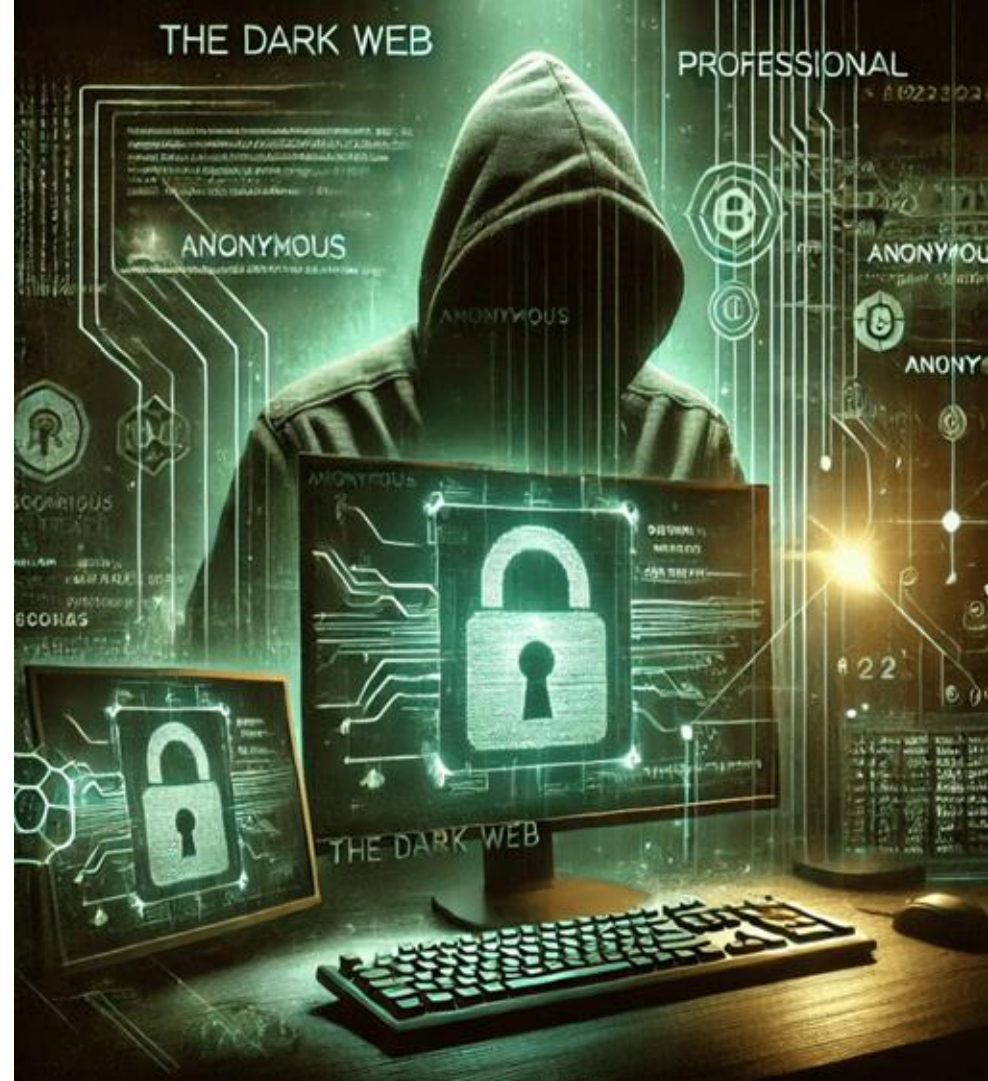




Predicting Darknet using Big Data Analytics



Team



Brayden Cloutier

Data Cleaning and
Pipelining



Bharathi K

Data Visualization &
Feature Selection



Mason Turner

Machine Learning

Agenda

1. Introduction
2. Objective
3. Dataset Summary
4. Data Flow Pipeline
5. Methodology
6. Results
7. Challenges
8. Future Work



Introduction

What?

- Darknet is a separate part of deep web where users surf around anonymously and in encrypted form.
- Darknet is often associated with illegal activities as it is tough to track the activities in darknet by authorities.

Use

Legitimate uses:- privacy for journalists and whistleblowers
- educational or research seeking anonymity

Illicit uses:- Illegal marketplaces
- Forums and Communities for illicit topics or terrorism
- Fraud and Financial crime spaces (selling stolen data)

Objective



Predicting darknet activity using dark-net related signals from large-scale data



PySpark, Databricks & ML libraries



Multi-layer Classification ML: Decision Tree, Logistic Regression, Random Forest, XGBoost, Gradient Boosted Tree

Dataset Summary

About

- Sourced from a repository for cybersecurity data
- 158,616 rows and 85 columns

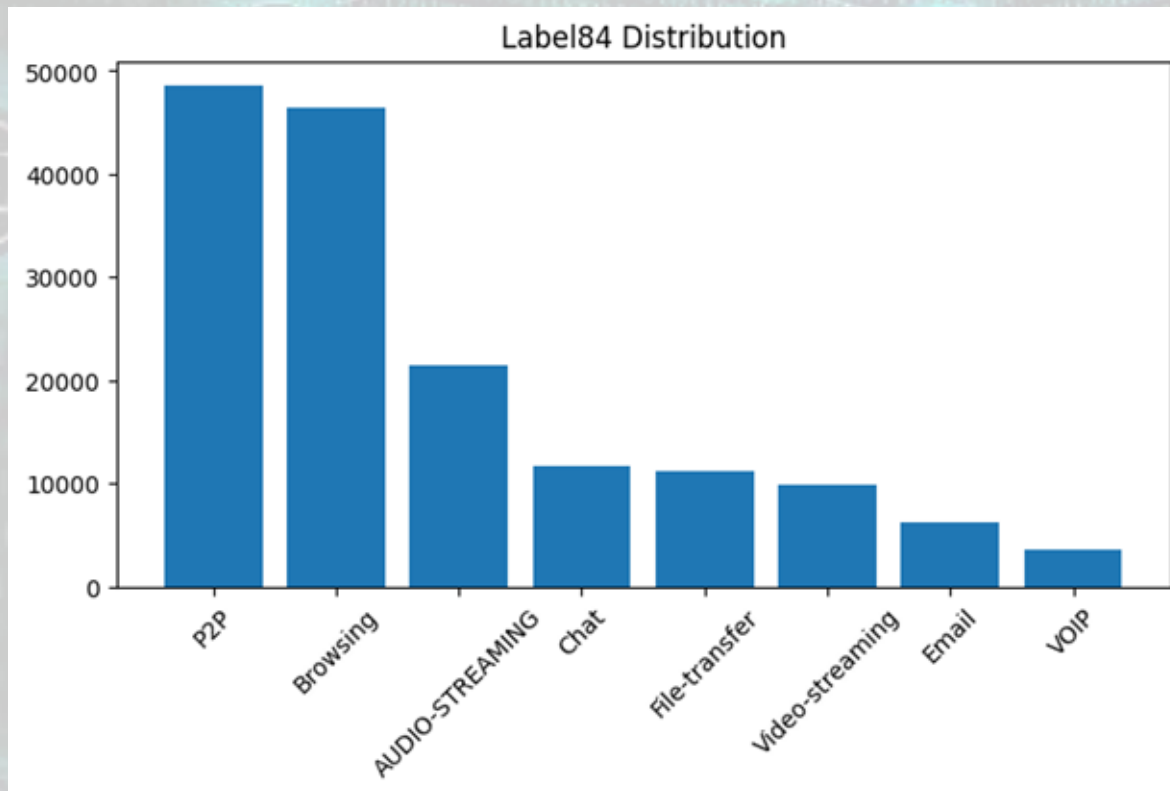
Types

- 83 numerical features
- 2 target columns
- Target column: Non-Tor, Tor, Non-VPN, VPN

Class

- Data had a column to identify the type of web.
- Audio Streaming, Browsing, Chat, Email, File Transfer, P2P, Video Streaming, VOIP.

Type Distribution



Data Preprocessing

Dropped Data

- High cardinality: 'Flow ID', 'Src IP', 'Timestamp', 'Dst IP'
- No Value : 'URG Flag Count', 'Bwd Psh Flags'

Split & Classify Data

- Labelled Non-Tor & Non-VPN as non- anomalous, Tor & VPN as anomalous.
- Anomalous – 2396 rows vs Non – Anomalous – 13488
- Split the dataset into 8 according to data types for training after setting aside some data for test.

Classification with multiple ML models

Size

- In total, we have 9 ML models in play
- While most computers can run a model, multiple could strain resources

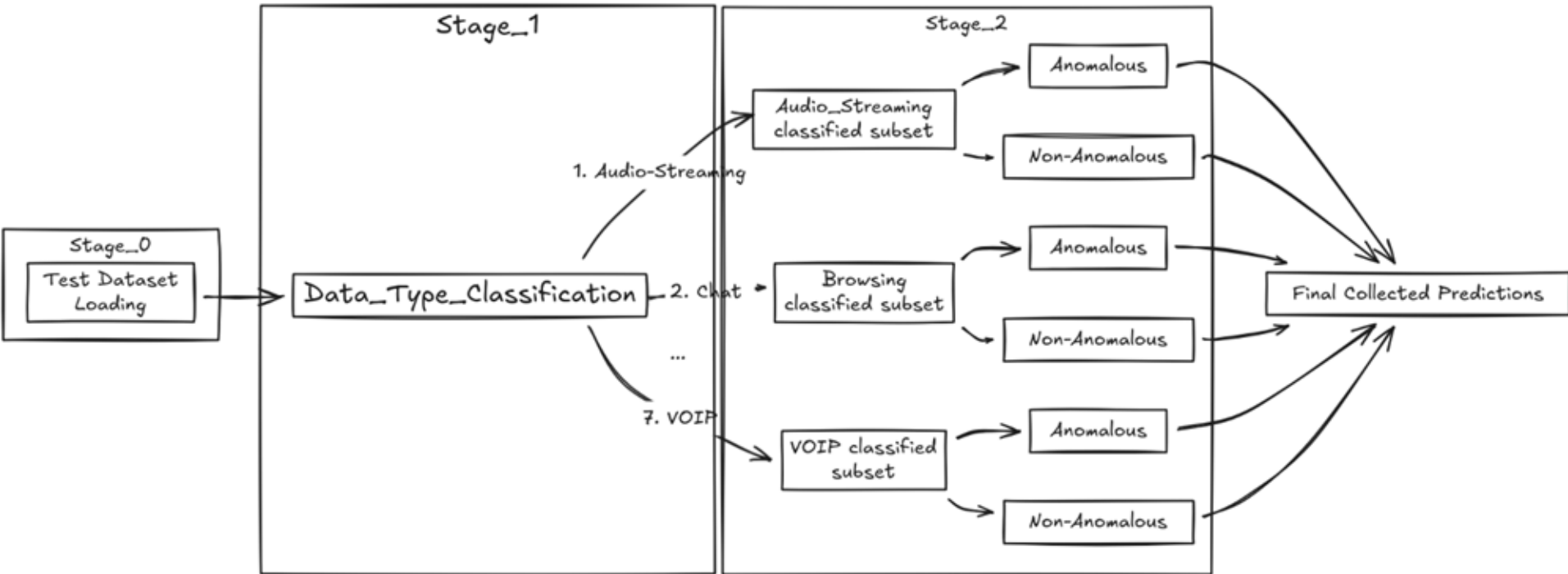
Strain

- Possible solution: For each datapoint, load in the right model
- Issue: Loading and unloading models every single datapoint is slow and inefficient

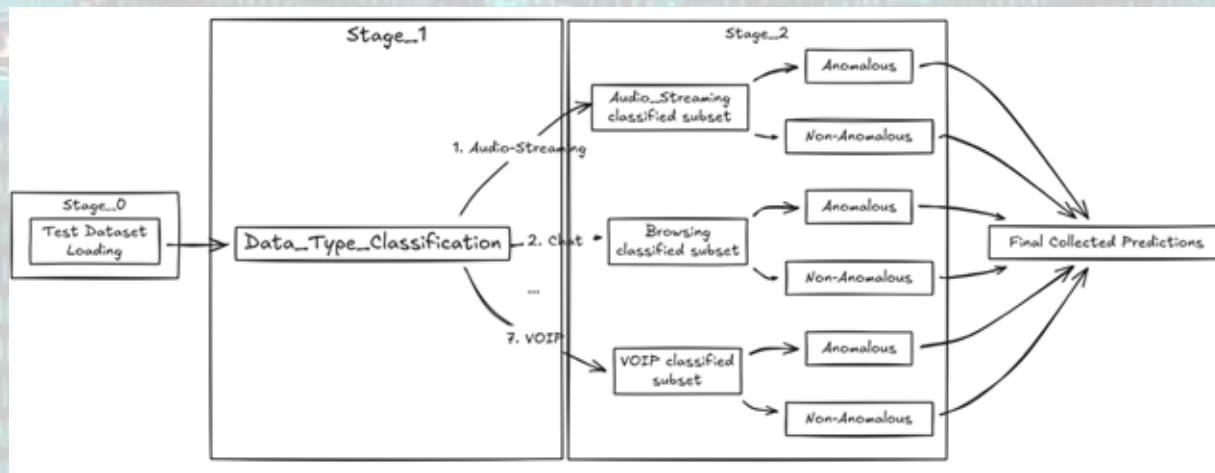
Solution

- Separate the entire dataset ahead of time by datatype
- Load each model once, and run all matching classifications

Data Flow: Pipeline – Multistage Classification

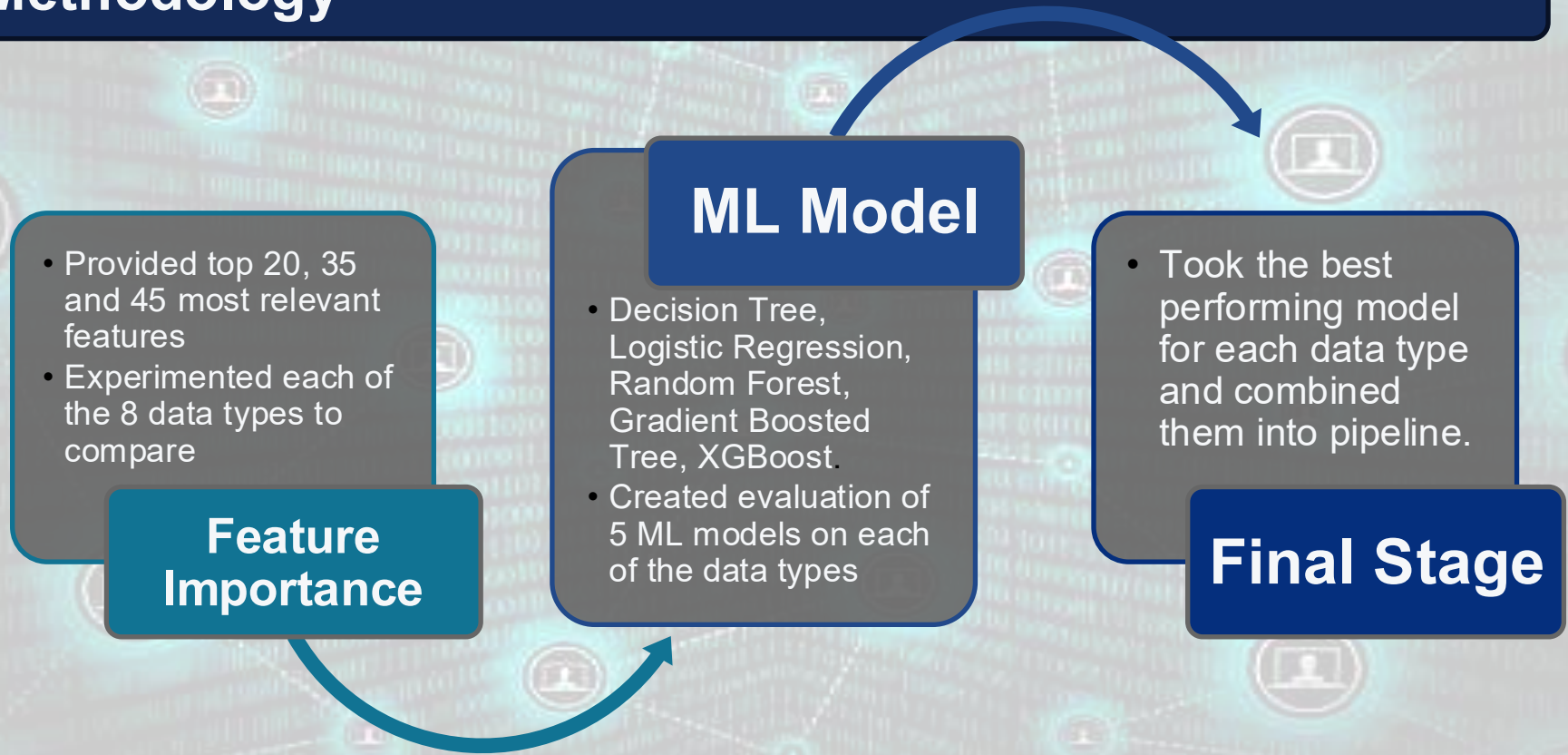


The Schema: great for durability, terrible for debugging



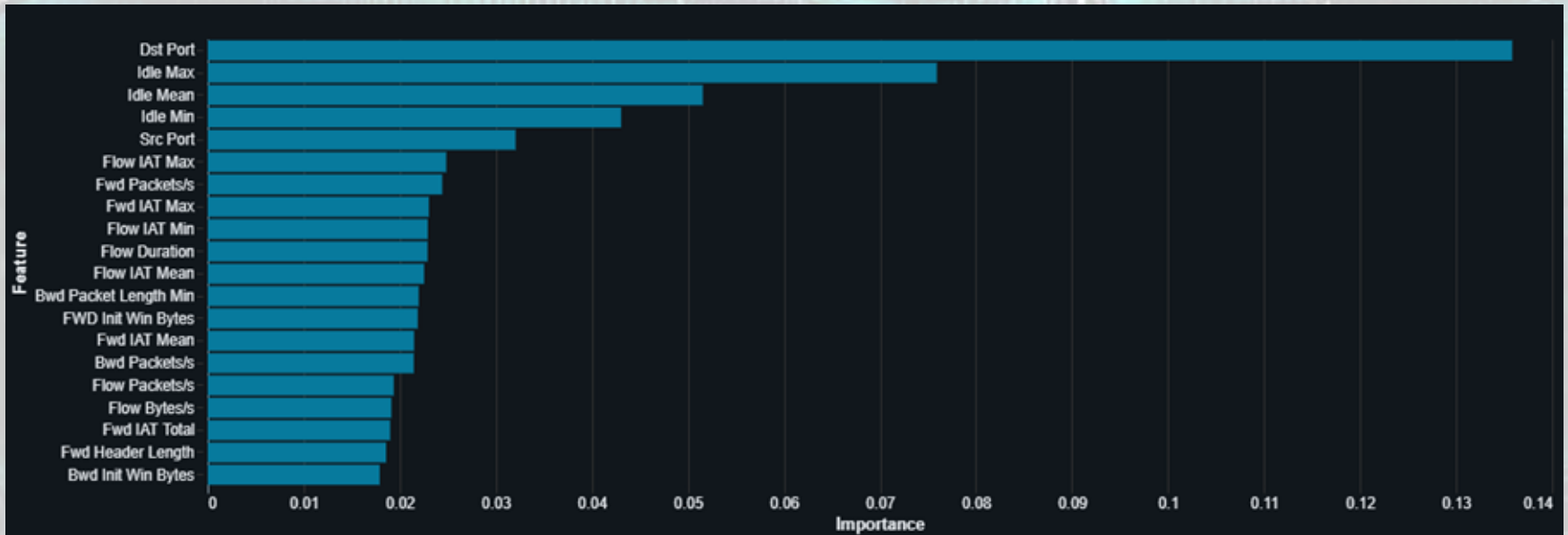
- Our dataset has string labels
- Pyspark models only work in doubles, and force output column to doubletype even if it predicts a string
- Each binary model string indexes individually
- Index > Stage Classification > Reverse Index

Methodology



Feature Importance

- Used Random Forest Classifier for Feature importance
- Did feature importance for all 8 of the data types and provided top 20, 35, 45 most relevant features for machine learning models.



Feature Importance – Comparison of top 20

- **Audio** : Average Packet Size, Packet Length Mean
- **Browsing**: Fwd Act Data Pkts, SYN Flag Count, Fwd PSH Flags, Flow IAT Std
- **Email**: RST Flag Count, ACK Flag Count
- **File-Transfer**: Packet Length Min, Fwd Packet Length Min
- **P2P**: Fwd Packet Length Std, Bwd Packet/Bulk Avg, Down/Up Ratio
- **Video**: Bwd IAT Max, Bwd IAT Mean, Bwd Header Length

- **Flow Duration except for Chat, File-Transfer & P2P**
- **Idle Max except for Browsing, Email & P2P**
- **Flow IAT Min**
- **Src Port**

Evaluation

1

Accuracy

Most simple evaluation of model's performance. Measures how many binary columns were classified correctly.

2

Precision

True Positive / (True Positive + False Positive) - Of all the instances classified as positive, how many were actually positive?

3

Recall

True Positive / (True Positive + False Negative) - How many of the positive instances were actually classified as positive?

4

F1-Score

Harmonic mean of Precision and Recall. Calculated as $F1 = 2 * (Precision * Recall) / (Precision + Recall)$.

Result -1st Stage

- XGBoost was the best performing model for every data type except for Browsing.
- The set of 20 was the best performing for Audio, Email and VOIP. The set of 35 was the best performing for Browsing, Data Type and P2P. The set of 45 was the best performing for Chat, File Transfer and Video Streaming.

Evaluation	Audio	Browsing	Chat	Email	File Transfer	P2P	Video	VOIP
Accuracy	98.33%	100%	99.68%	99.47%	99.66%	100%	96.54%	85.80%
Precision	98.33%	100%	99.68%	99.47%	99.66%	100%	96.56%	86.02%
Recall	98.33%	100%	99.68%	99.47%	99.66%	100%	96.54%	85.80%
F1-Score	98.33%	100%	99.68%	99.47%	99.66%	100%	96.54%	85.82%

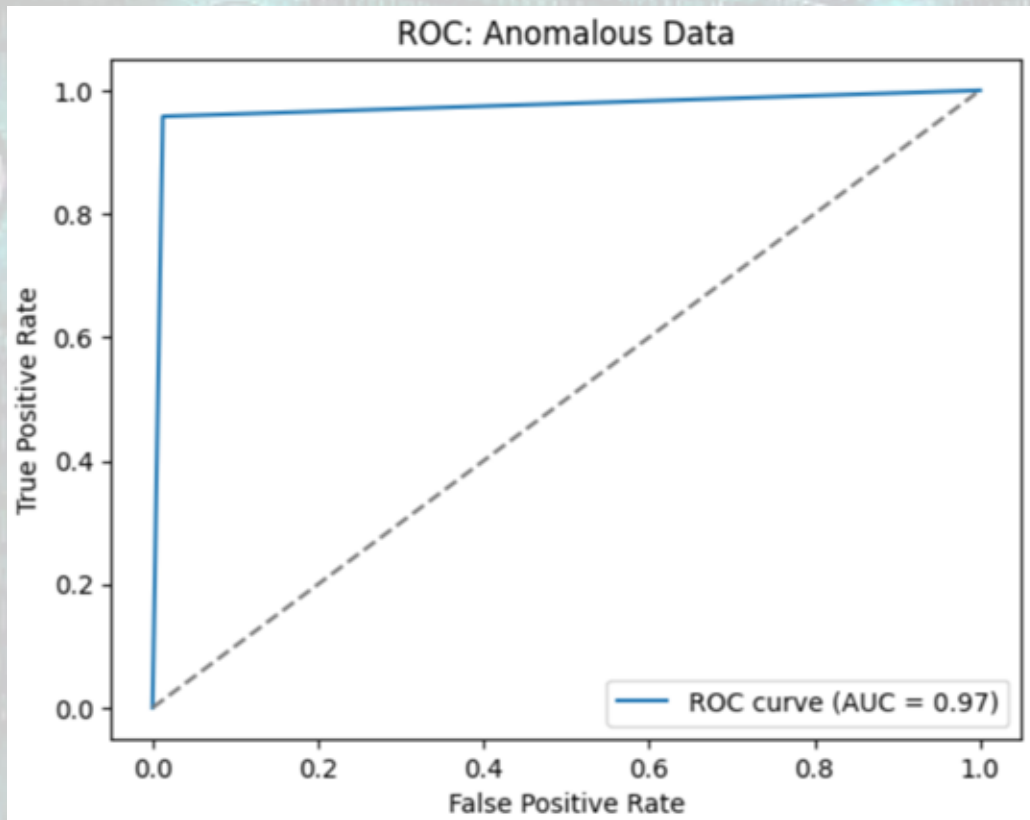
Result - 2nd Stage

- XGBoost was the best performing model for data type classification.

2nd Stage - Results

- Accuracy: 92.35%
- Precision: 92.18%
- Recall: 92.35%
- F1-Score: 92.22%

Result – Overall Model



Overall - Results

- Accuracy: 98.31%
- Precision: 98.34%
- Recall: 98.31%
- F1-Score: 98.32%

Challenges

Strict Schemas

If I wasn't explicit with casting, both errors and damaged results could occur.

Inconsistent Error Checking

Loading models in sequence sometimes caused internal errors in the cluster.

Model loading

It just worked. I told it to load, I told it to transform the data, no complaints there.

Partitioning

Due to the nature of the task, the flow of the code split well into separate functions.

How do I handle Class Balancing? Check which Label is more common for each data type, then randomly select an identical number of rows from the larger as there are rows in the smaller.

Future Work

- Utilizing Neural Networks in addition to the machine learning models we tested to see if detection can be improved.
- Getting a baseline: We only did a 2 stage ML methodology. How does it stack up against one monolithic ML model? What about a classification focused LLM?
- Real-Time Detection: enable near real-time monitoring of darknet activity signals



Thank You